arXiv:1201.0306v1 [stat.CO] 31 Dec 2011

Alternating Linearization for Structured Regularization Problems

Xiaodong Lin* Minh Pham† Andrzej Ruszczyński‡

December 31, 2011

Abstract

We adapt the alternating linearization method for proximal decomposition to structured regularization problems, in particular, to the generalized lasso problems. The method is related to two well-known operator splitting methods, the Douglas–Rachford and the Peaceman–Rachford method, but it has descent properties with respect to the objective function. Its convergence mechanism is related to that of bundle methods of nonsmooth optimization. We also discuss implementation for very large problems, with the use of specialized algorithms and sparse data structures. Finally, we present numerical results for several synthetic and real-world examples, including a three-dimensional fused lasso problem, which illustrate the scalability, efficacy, and accuracy of the method.

Keywords: lasso, fused lasso, nonsmooth optimization, operator splitting

*Department of Management Science and Information Systems, Rutgers University, 94 Rockefeller Rd., Piscataway, NJ 08854; Email: lin@business.rutgers.edu

†Rutgers Center for Operations Research (RUTCOR), Rutgers University, 640 Bartholomew Rd., Piscataway, NJ 08854; Email: ptuanminh@gmail.com

‡Department of Management Science and Information Systems, Rutgers University, 94 Rockefeller Rd., Piscataway, NJ 08854; Email: rusz@business.rutgers.edu

1 Introduction

Regularization techniques that encourage sparsity in parameter estimation have gained increasing popularity recently. The most widely used example is *lasso* (Tibshirani, 1996), where the loss function $f(\cdot)$ is penalized by the ℓ_1-norm of the unknown coefficients $\beta \in \mathbb{R}^p$, to form a modified objective function,

$$\mathcal{L}(\beta) = f(\beta) + \lambda \|\beta\|_1, \quad \lambda > 0, \tag{1}$$

in order to shrink irrelevant coefficients to zero. Many efficient algorithms have been proposed to solve this problem, including (Fu, 1998; Daubechies et al., 2004; Efron et al., 2004) and (Friedman et al., 2007). Some of them are capable of handling massive data sets with tens of thousands of variables and observations.

For many practical applications, physical constraints and domain knowledge may mandate additional structural constraints on the parameters. For example, in cancer research, it may be important to consider groups of interacting genes in each pathway rather than individual genes. In image analysis, it is natural to regulate the differences between neighboring pixels in order to achieve smoothness and reduce noise. In light of these popular demands, a variety of structured penalties have been proposed to incorporate prior information. One of the most important structural penalties is the *fused lasso* proposed in (Tibshirani et al., 2005). It utilizes the natural ordering of input variables to achieve parsimonious parameter estimation on neighboring coefficients. Beck and Teboulle (2009) adopt the *total variation penalty* for image denoising and deblurring, in a similar fashion to the two-dimensional fused lasso. Chen et al. (2010) proposed the *graph induced fused lasso* that penalizes differences between coefficients associated with nodes that are connected. A more general structural lasso framework was proposed in (Tibshirani and Taylor, 2011), with the following form:

$$\mathcal{L}(\beta) = f(\beta) + \lambda \|R\beta\|_1, \quad \lambda > 0, \tag{2}$$

where R is an $m \times p$ matrix that defines the structural constraints one wants to impose on the coefficients. Many regularization problems, including fused lasso and graph induced fused lasso, can be cast in this framework.

When the structural matrix R is relatively simple, as in the original lasso case with $R = I$, traditional path algorithms and coordinate descent techniques can be used to solve the optimization problem efficiently (Friedman et al., 2007). For more complex structural regularization, these methods cannot be directly applied. One of the key difficulties is the non-separability of the nonsmooth penalty function. Coordinate descent methods fail to converge under this circumstances (Tseng, 2001). Generic solvers, such as interior point methods, can sometimes be used; unfortunately they become increasingly inefficient for large size problems, particularly when the design matrix is ill-conditioned (Chen et al., 2011).

In the past two years, many efforts have been devoted to developing efficient optimization techniques for solving regularization problems using structured penalties. Liu et al. (2010) developed a first order and a split Bregman scheme, respectively, for solving similar class of problems. In many practical studies, the convergence of these two methods can not be guaranteed. Chen et al. (2011) proposed a modified proximal technique for the general structurally penalized problems. It is based on a first order approximation of the

nonsmooth penalty function, which can become unstable when dimension is high. Meanwhile, several path algorithms have also been proposed to compute the whole regularization path for the general fused lasso problem. Hoefling (2010) developed a path algorithm for solving (2) when the matrix $X^T X$ is nonsingular. This technique is not applicable to cases with large dimension of β and small number of observations, such as gene expression and brain imaging analysis. Tibshirani and Taylor (2011) extended the path algorithm to include all design matrices X, by computing the regularization path of the dual problem. Although fairly general, this version of the path algorithm does not scale well with data dimension, as the knots of the piecewise linear solution path become very dense. Many of the proposed approaches are versions of the *operator splitting methods* or their dual versions, *alternating direction methods* (see, e.g., Boyd et al. (2010); Combettes and Pesquet (2010), and the references therein). Although fairly general and universal, they frequently suffer from slow tail convergence (see (He and Yuan, 2011) and the references therein).

Thus, a need arises to develop a general approach that can solve large scale structured regularization problem efficiently. For such an approach to be successful in practice, it should guarantee to converge at a fast rate, be able to handle massive data sets, and should not rely on approximating the penalty function. In this paper, we propose a framework based on the alternating linearization algorithm of (Kiwiel et al., 1999), that satisfies all these requirements.

Formally, we write the objective function as a sum of two convex functions,

$$\mathcal{L}(\beta) = f(\beta) + h(\beta), \tag{3}$$

where $f(\beta)$ is a loss function, which is assumed to be convex with respect to β, and $h(\cdot)$ is a convex penalty function. Any of the functions (or both) may be nonsmooth, but an essential requirement of our framework is that each of them can be easily minimized with respect to β, when augmented by a linear-quadratic term $\sum_{i=1}^{p} \left(s_i\beta_i + d_i\beta_i^2\right)$, with some vectors $s, d \in \mathbb{R}^p$, $d > 0$. Our method bears resemblance to operator splitting and alternating direction approaches, but differs from them in the fact that it is *monotonic* with respect to the values of (3). We discuss these relations and differences later in section 2.2, but roughly speaking, a special test applied at every iteration of the method decides which of the operator splitting iterations is the most beneficial one.

In our applications, we focus on the quadratic loss function $f(\cdot)$ and the penalty function in the form of generalized lasso (2), as the most important case, where comparison with other approaches is available. This case satisfies the requirement specified above, and allows for substantial specialization and acceleration of the general framework of alternating linearization. In fact, it will be clear from our presentation that any convex loss function $f(\cdot)$ can be handled in exactly the same way.

An important feature of our approach is that problems with the identity design matrix are solved exactly in one iteration, even for very large dimension.

The remainder of the paper is organized as follows. In Section 2, we introduce the alternating linearization method and we discuss its relations to other approaches. Section 3 briefly discusses the application to lasso problems. In section 4 we describe the application to generalized lasso problems. Section 5 presents simulation results and real data examples, which illustrate the efficacy, accuracy, and scalability of the alternating

linearization method. Concluding remarks are presented in section 6. The appendix contains details about the algorithms used to solve the subproblems of the alternating linearization method.

2 The alternating linearization method

2.1 Outline of the method

In this section, we describe the alternating linearization (ALIN) approach to minimize (3). It is an iterative method, which generates a sequence of approximations $\{\hat{\beta}^k\}$ converging to a solution of the original problem (3), and two auxiliary sequences: $\{\tilde{\beta}_h^k\}$ and $\{\tilde{\beta}_f^k\}$, where k is the iteration number. Each iteration of the ALIN algorithm consists of solving two subproblems: the h-*subproblem* and the f-*subproblem*, and of an *update step*, applied after any of the subproblems, or after each of them.

At the beginning we set $\tilde{\beta}_f^0 = \hat{\beta}^0$, where $\hat{\beta}^0$ is the starting point of the method. In the description below, we suppress the superscript k denoting the iteration number, to simplify notation.

The h-subproblem

We linearize $f(\cdot)$ at $\tilde{\beta}_f$, and approximate it by the function

$$\tilde{f}(\beta) = f(\tilde{\beta}_f) + s_f^T(\beta - \tilde{\beta}_f).$$

If $f(\cdot)$ is differentiable, then $s_f = \nabla f(\tilde{\beta}_f)$; for a general convex $f(\cdot)$, we select a subgradient $s_f \in \partial f(\tilde{\beta}_f)$. In the first iteration, this may be an arbitrary subgradient; at later iterations special selection rules apply, as described in (7) below.

The approximation is used in the optimization problem

$$\min_{\beta} \ \tilde{f}(\beta) + h(\beta) + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2, \tag{4}$$

in which the last term is defined as follows:

$$\|\beta - \hat{\beta}\|_D^2 = (\beta - \hat{\beta})^T D(\beta - \hat{\beta}),$$

with a diagonal matrix $D = \text{diag}\{d_j,\ j = 1, \ldots, p\}, d_j > 0, j = 1, \ldots, p$. The solution of the h-subproblem (4) is denoted by $\tilde{\beta}_h$.

We complete this stage by calculating the subgradient of $h(\cdot)$ at $\tilde{\beta}_h$, which features in the optimality condition for the minimum in (4):

$$0 \in s_f + \partial h(\tilde{\beta}_h) + D(\tilde{\beta}_h - \hat{\beta}).$$

Elementary calculation yields the right subgradient $s_h \in \partial h(\tilde{\beta}_h)$:

$$s_h = -s_f - D(\tilde{\beta}_h - \hat{\beta}). \tag{5}$$

The f-subproblem

Using the subgradient s_h we construct a linear minorant of the penalty function $h(\cdot)$ as follows:

$$\tilde{h}(\beta) = h(\tilde{\beta}_h) + s_h^T(\beta - \tilde{\beta}_h).$$

This approximation is employed in the optimization problem

$$\min_{\beta} \; f(\beta) + \tilde{h}(\beta) + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2. \tag{6}$$

The optimal solution of this problem is denoted by $\tilde{\beta}_f$. It will be used in the next iteration as the point at which the new linearization of $f(\cdot)$ will be constructed. The next subgradient of $f(\cdot)$ to be used in the h-subproblem will be

$$s_f = -s_h - D(\tilde{\beta}_f - \hat{\beta}). \tag{7}$$

The update step

The update step can be applied after any of the subproblems, or after both of them. It changes the current best approximation of the solution $\hat{\beta}$, if certain improvement conditions are satisfied. It uses a parameter $\gamma \in (0, 1)$. We describe it here for the case of applying the update step after the f-subproblem; analogous operations are carried out if the update step is applied after the h-subproblem.

At the beginning of the update step the stopping criterion is verified. If

$$f(\tilde{\beta}_f) + \tilde{h}(\tilde{\beta}_f) \geq f(\hat{\beta}) + h(\hat{\beta}) - \varepsilon, \tag{8}$$

the algorithm terminates. Here $\varepsilon > 0$ is the stopping test parameter.

If the the stopping test is not satisfied, we check the inequality

$$f(\tilde{\beta}_f) + h(\tilde{\beta}_f) \leq (1-\gamma)\big[f(\hat{\beta}) + h(\hat{\beta})\big] + \gamma\big[f(\tilde{\beta}_f) + \tilde{h}(\tilde{\beta}_f)\big]. \tag{9}$$

If it is satisfied, then we update $\hat{\beta} \leftarrow \tilde{\beta}_f$; otherwise $\hat{\beta}$ remains unchanged.

If the update step is applied after the h-subproblem, we use $\tilde{\beta}_h$ instead if $\tilde{\beta}_f$ in the inequalities (8) and (9).

The update step is a crucial component of the alternating linearization algorithm; it guarantees that the sequence $\{\mathcal{L}(\hat{\beta}^k)\}$ is monotonic, and it stabilizes the entire algorithm (see the remarks at the end of section 5.1).

2.2 Relation to operator splitting and alternating direction methods

Our approach is intimately related to *operator splitting methods* and their dual versions, *alternating direction methods*, which are recently very popular in the area of signal processing (see, e.g., (Boyd et al., 2010; Combettes and Pesquet, 2010; Fadili and Peyré, 2011)). To discuss these relations, it is convenient to present

our method formally, and to introduce two running *proximal centers*:

$$z_f = \hat{\beta} - D^{-1} s_f,$$
$$z_h = \hat{\beta} - D^{-1} s_h.$$

After elementary manipulations we can absorb the linear terms into the quadratic terms and summarize the alternating linearization method as follows.

Algorithm 1 Alternating Linearization

1: **repeat**
2: $\tilde{\beta}_h \leftarrow \arg\min \left\{ h(\beta) + \frac{1}{2}\|\beta - z_f\|_D^2 \right\}$
3: **if** (*Update Test for* $\tilde{\beta}_h$) **then**
4: $\hat{\beta} \leftarrow \tilde{\beta}_h$
5: **end if**
6: $z_h \leftarrow \hat{\beta} + \tilde{\beta}_h - z_f$
7: $\tilde{\beta}_f \leftarrow \arg\min \left\{ f(\beta) + \frac{1}{2}\|\beta - z_h\|_D^2 \right\}$
8: **if** (*Update Test for* $\tilde{\beta}_f$) **then**
9: $\hat{\beta} \leftarrow \tilde{\beta}_f$
10: **end if**
11: $z_f \leftarrow \hat{\beta} + \tilde{\beta}_f - z_h$
12: **until** (*Stopping Test*)

The *Update Test* in lines 3 and 8 is the corresponding version of inequality (9). The *Stopping Test* is inequality (8).

If we assume that the update steps in lines 4 and 9 are carried out after *every* h-subproblem and *every* f-subproblem, without verifying the update test (9), then the method becomes equivalent to a scaled version of the Peaceman–Rachford algorithm (originally proposed by (Peaceman and Rachford, 1955) for PDEs and later generalized and analyzed by (Lions and Mercier, 1979); see also (Combettes, 2009) and the references therein). If $D = \rho I$ with $\rho > 0$, then we obtain an unscaled version of this algorithm.

If we assume that the update steps are carried out after *every* h-subproblem without verifying inequality (9), but *never* after f-subproblems, then the method becomes equivalent to a scaled version of the Douglas–Rachford algorithm (introduced by (Douglas and Rachford, 1956), and generalized and analyzed by (Lions and Mercier, 1979); see also (Bauschke and Combettes, 2011) and the references therein). As the roles of f and h can be switched, the method in which updates are carried always after f-subproblems, but never after h-subproblems, is also equivalent to a scaled Douglas–Rachford method.

Operator splitting methods are not monotonic with respect to the values of the objective function $\mathcal{L}(\beta)$. Their convergence is based on monotonicity with respect to the distance to the optimal solution of the problem (Lions and Mercier, 1979; Eckstein and Bertsekas, 1992).

In contrast, the convergence mechanism of our method is different; it draws from some ideas of bundle methods in nonsmooth optimization (Hiriart-Urruty and Lemaréchal, 1993; Kiwiel, 1985; Ruszczyński, 2006). Its key element is the update test employed in (9). At every iteration we decide whether it is beneficial

to make a Peaceman–Rachford step, any of the two possible Douglas–Rachford steps, or none. In the latter case, which we call the *null step*, $\bar{\beta}$ remains unchanged, but the trial points $\tilde{\beta}_h$ and $\tilde{\beta}_f$ are updated. These updates continue, until $\tilde{\beta}_h$ or $\tilde{\beta}_f$ become better than $\hat{\beta}$, or until optimality is detected (*cf.* the remarks at the end of section 5.1).

Alternating direction methods are dual versions of the operator splitting methods, applied to the following equivalent form of the problem of minimizing (3):

$$\min f(\beta_1) + h(\beta_2), \quad \text{subject to} \quad \beta_1 = \beta_2. \tag{10}$$

In regularized signal processing problems, when $f(\beta) = \varphi(X\beta)$ with some fixed matrix X, the convenient problem formulation is

$$\min \varphi(v) + h(\beta), \quad \text{subject to} \quad v = X\beta.$$

The dual functional has the form of a sum of two functions, and the operator splitting methods apply. The reader may consult (Boyd et al., 2010; Combettes and Pesquet, 2010) for appropriate derivations. It is also worth mentioning that the alternating direction methods are sometimes called *split Bregman methods* in the signal processing literature (see, e.g., (Goldstein and Osher, 2009; Ye and Xie, 2011), and the references therein).

However, to apply our alternating linearization method to the dual problem, we would have to be able to quickly compute the value of the dual function, in order to verify the update condition (9), as discussed in detail in (Kiwiel et al., 1999). This is the reason for our preference toward the primal version of the method.

2.3 Convergence

Convergence properties of the alternating linearization method follow from the general theory developed in (Kiwiel et al., 1999). Indeed, after the change of variables $\xi = D^{1/2}\beta$ we see that the method is identical to Algorithm 3.1 of (Kiwiel et al., 1999), with $\rho_k = 1$. The following statement is a direct consequence of (Kiwiel et al., 1999, Theorem 4.8).

Theorem 1 *Suppose that the set of minima of the function* (3) *is nonempty. Then the sequence* $\{\hat{\beta}^k\}$ *generated by the algorithm is convergent to a minimum point* β^* *of the function* (3)*. Moreover, every accumulation point* (s_f^*, s_h^*) *of the sequence* $\{(s_f^k, s_h^k)\}$ *satisfies the relations:* $s_f^* \in \partial f(\beta^*)$*,* $s_h^* \in \partial h(\beta^*)$*, and* $s_f^* + s_h^* = 0$*.*

For structured regularization problems the assumption of the theorem is satisfied, because both the loss function $f(\cdot)$ and the regularizing function $h(\cdot)$ are bounded from below, and one of the purposes of the regularization term is to make the set of minima of the function $\mathcal{L}(\cdot)$ nonempty and bounded.

3 Application to lasso regression

First, we demonstrate the alternating linearization algorithm (ALIN) on the classical lasso regression problem. Due to the separable nature of the penalty function, very efficient coordinate descent methods are applicable to this problem as well (Tseng, 2001), but we wish to illustrate our approach on the simplest case first.

In the lasso regression problem we have

$$f(\beta) = \tfrac{1}{2}\|y - X\beta\|_2^2, \qquad h(\beta) = \lambda\|\beta\|_1,$$

where X is the $n \times p$ design matrix, $y \in \mathbb{R}^n$ is the vector of response variables, $\beta \in \mathbb{R}^p$ is the vector of regression coefficients, and $\lambda > 0$ is a parameter of the model.

We found it essential to use $D = \operatorname{diag}(X^T X)$, that is, $d_j = X_j^T X_j$, $j = 1, \ldots, p$. This choice is related to the *diagonal quadratic approximation* of the function $f(\beta) = \frac{1}{2}\|y - X\beta\|_2^2$, which was employed (for similar objectives in the context of augmented Lagrangian minimization) by Ruszczyński (1995). Indeed, in the h-subproblem in the formula (11) below, the quadratic regularization term is a quadratic form built on the diagonal of the Hessian of $f(\cdot)$.

The h-subproblem

The problem (4), after skipping constants, simplifies to the following form

$$\min_{\beta} \; s_f^T \beta + \lambda\|\beta\|_1 + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2. \tag{11}$$

with $s_f = X^T(X\tilde{\beta}_f - y)$. Writing $\tau_j = \hat{\beta} - \tilde{s}_{fj}/d_j$, we obtain the following closed form solutions of (11), which can be calculated component-wise:

$$\tilde{\beta}_{hj} = \operatorname{sgn}(\tau_j) \max\Big(0, |\tau_j| - \frac{\lambda}{d_j}\Big), \quad j = 1, \ldots, p. \tag{12}$$

The subgradient s_h of $h(\cdot)$ at $\tilde{\beta}_h$ is calculated by (5).

The f-subproblem

The problem (6), after skipping constants, simplifies to the unconstrained quadratic programming problem

$$\min_{\beta} \; s_h^T \beta + \tfrac{1}{2}\|y - X\beta\|_2^2 + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2. \tag{13}$$

Its solution can be obtained by solving the following symmetric linear system in $\delta = \beta - \hat{\beta}$:

$$(X^T X + D)\delta = X^T(y - X\hat{\beta}) - s_h. \tag{14}$$

This system can be efficiently solved by the preconditioned conjugate gradient method (see, e.g., (Golub and Van Loan, 1996)), with the diagonal preconditioner $2D = 2\operatorname{diag}(X^T X)$. Its application does not require the explicit form of the matrix $X^T X$; only matrix-vector multiplications with X and X^T are employed, and they can be implemented with sparse data structures.

4 Application to generalized lasso regression

In the following we apply the alternating linearization algorithm to solve the generalized lasso problem (2). Here we assume the least squares loss, as in the previous subsection. The objective function can be written as follows:

$$\mathcal{L}(\beta) = f(\beta) + h(\beta) = \tfrac{1}{2}\|y - X\beta\|_2^2 + \lambda\|R\beta\|_1. \tag{15}$$

For example, for the one-dimensional fused lasso, R is the following $(p-1) \times p$ matrix:

$$R = \begin{bmatrix} -1 & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & -1 & 1 \end{bmatrix},$$

but our derivations are valid for any form of R.

The h-subproblem

The h-subproblem can be equivalently formulated as follows:

$$\min_{\beta,z} \; s_f^T\beta + \lambda\|z\|_1 + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2 \quad \text{subject to} \quad R\beta = z. \tag{16}$$

Owing to the use of $D = \text{diag}(X^TX)$, and with $s_f = X^T(X\hat{\beta} - y)$, it is a quite accurate approximation of the original problem, especially for sparse X (Ruszczyński, 1995).

The Lagrangian of problem (16) has the form

$$L(\beta, z, \mu) = s_f^T\beta + \lambda\|z\|_1 + \mu^T(R\beta - z) + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2,$$

where μ is the dual variable. We see that the minimum of the Lagrangian with respect to z is finite if and only if $\|\mu\|_\infty \leq \lambda$. Under this condition, the minimum value of the z-terms is zero and we can eliminate them from the Lagrangian. We arrive to its reduced form,

$$\hat{L}(\beta, \mu) = s_f^T\beta + \mu^T R\beta + \tfrac{1}{2}\|\beta - \hat{\beta}\|_D^2. \tag{17}$$

To calculate the dual function, we minimize $\hat{L}(\beta, \mu)$ over $\beta \in \mathbb{R}^p$. After elementary calculations, we obtain the solution

$$\tilde{\beta}_h = \hat{\beta} - D^{-1}(s_f + R^T\mu). \tag{18}$$

Substituting it back to (17), we arrive to the following dual problem:

$$\max_{\mu} \; -\tfrac{1}{2}\mu^T RD^{-1}R^T\mu + \mu^T R(\hat{\beta} - D^{-1}s_f) \quad \text{subject to} \quad \|\mu\|_\infty \leq \lambda. \tag{19}$$

This is a box-constrained quadratic programming problem. It can be efficiently solved by the active-set

box-constrained preconditioned conjugate gradient algorithm with spectral projected gradients, as described in (Birgin and Martínez, 2002; Friedlander and Martínez, 1994). The diagonal of the matrix $RD^{-1}R^T$ is a good preconditioner for this method in the applications that we dealt with. We summarize the most important operations of this method in the Appendix. It should be stressed that its application does not require the explicit form of the matrix $RD^{-1}R^T$; only matrix-vector multiplications with R and R^T are employed, and they can be implemented with sparse data structures.

The solution $\tilde{\mu}$ of the dual problem can be substituted into (18) to obtain the primal solution.

The f-subproblem

We obtain the update $\tilde{\beta}_f$ by solving the linear equation system (14), exactly as in the lasso case.

The special case of $X = I$

If the design matrix $X = I$ in (15), then our method solves the problem in one iteration, when started from $\hat{\beta} = y$. Indeed, in this case we have $s_f = 0$, $D = I$, and the first h-subproblem becomes equivalent to the original problem (15):

$$\min_{\beta,z} \ \lambda\|z\|_1 + \tfrac{1}{2}\|\beta - y\|_2^2 \quad \text{subject to} \quad R\beta = z. \tag{20}$$

The dual problem (19) simplifies as follows:

$$\max_{\mu} \ -\tfrac{1}{2}\mu^T RR^T\mu + \mu^T Ry \quad \text{subject to} \quad \|\mu\|_\infty \leq \lambda. \tag{21}$$

It can be solved by the same conjugate gradient method with bounds, with the preconditioner equal to the diagonal of the matrix RR^T. The optimal primal solution is then $\tilde{\beta}_h = y - R^T\mu$.

5 Numerical experiments

In this section, we present results on a number of simulation and real data studies involving a variety of non differentiable penalty functions. We compare the alternating linearization algorithm (ALIN) with competing approaches in terms of iteration steps, computation time and estimation accuracy. All these studies are performed on an AMD 2.6GHZ, 4GB RAM computer using MATLAB.

5.1 Simulation studies

In this experiment, we compare the ALIN algorithm with three different approaches using data sets generated from a linear regression model $y = \sum_{j=1}^{p} x_j\beta_j + \epsilon$ with pre-specified coefficients β_j and varying dimension p. x_j is drawn from the normal distribution with zero mean and unit variance. ϵ is the noise vector that is generated from the normal distribution with zero mean and variance equal to 0.01. Among these coefficients,

10% equal 1, 20% equal 2, and the rest are zeros. For instance, with $p = 100$, we may have

$$\beta_j = \begin{cases} 1 & \text{for } j = 11, 12, \ldots, 20, \\ 2 & \text{for } j = 20, \ldots, 49, \\ 0 & \text{otherwise.} \end{cases}$$

Table 1 reports the run times of ALIN and three competing algorithms: the generic quadratic programming solver (SQOPT), the alternating direction (split Bregman) method (BREGMAN) of (Ye and Xie, 2011) and Nesterov's subgradient method (SLEP) of (Liu et al., 2011; Nesterov, 2007). In this study, we fix the sample size to 1000 and vary the dimension of the problem from 1000 to 50000. Each method is repeated 10 times and the average is reported. The Split Bregman method and SLEP are run to optimal solution corresponding to the stopping criteria built in the packages. We stop ALIN runs when the objective function value attained is as good as that of SLEP. Judging from these results, ALIN clearly outperforms the other methods in terms of speed for most cases. The improvements on run time range from 1.5 to 3 folds depending on the experimental setting, and becomes more significant when the data dimension grows higher. This is particularly significant in view of the fact that ALIN was implemented as a MATLAB code, as opposed to the executables in the other cases.

Table 1: Run time comparison between SQOPT, BREGMAN, SLEP and ALIN (in seconds).

Problem size	SQOPT	BREGMAN	SLEP	ALIN
$n = 1000$, $p = 100$, $\lambda = 0.1$	0.2	15.2	0.02	0.23
$n = 1000$, $p = 1000$, $\lambda = 0.1$	9.8	103	0.7	7
$n = 1000$, $p = 5000$, $\lambda = 0.1$	940	471	31	17.6
$n = 1000$, $p = 10000$, $\lambda = 0.1$	NA	879	126.7	92.7
$n = 1000$, $p = 20000$, $\lambda = 0.1$	NA	2133	489	433.5
$n = 1000$, $p = 50000$, $\lambda = 0.1$	NA	3633	1401.6	666.2

It is interesting to compare the approximation of the optimal solution found by the methods, as depicted in Figure 1.

Next we investigate how our method approaches the optimal objective function value compared to other methods. Using the above simulated data set with $n = 1000$, $p = 5000$, and $\lambda = 0.1$, we run three methods to convergence and plot the objective function value attained along the iterations. We obtain the optimal value $\mathcal{L}^*$ from SQOPT. At each iteration, we can calculate the difference between the optimal value and the current function value. The plots in Figure 2 demonstrate how far the methods are from the optimal value along the iterations. We also provide in Figure 3 the dependence of the running time of ALIN on the dimensions of the problem, to illustrate its scalability. The efficiency of the method is due mainly to its good convergence properties, but also to the efficiency of the preconditioned conjugate gradient method for

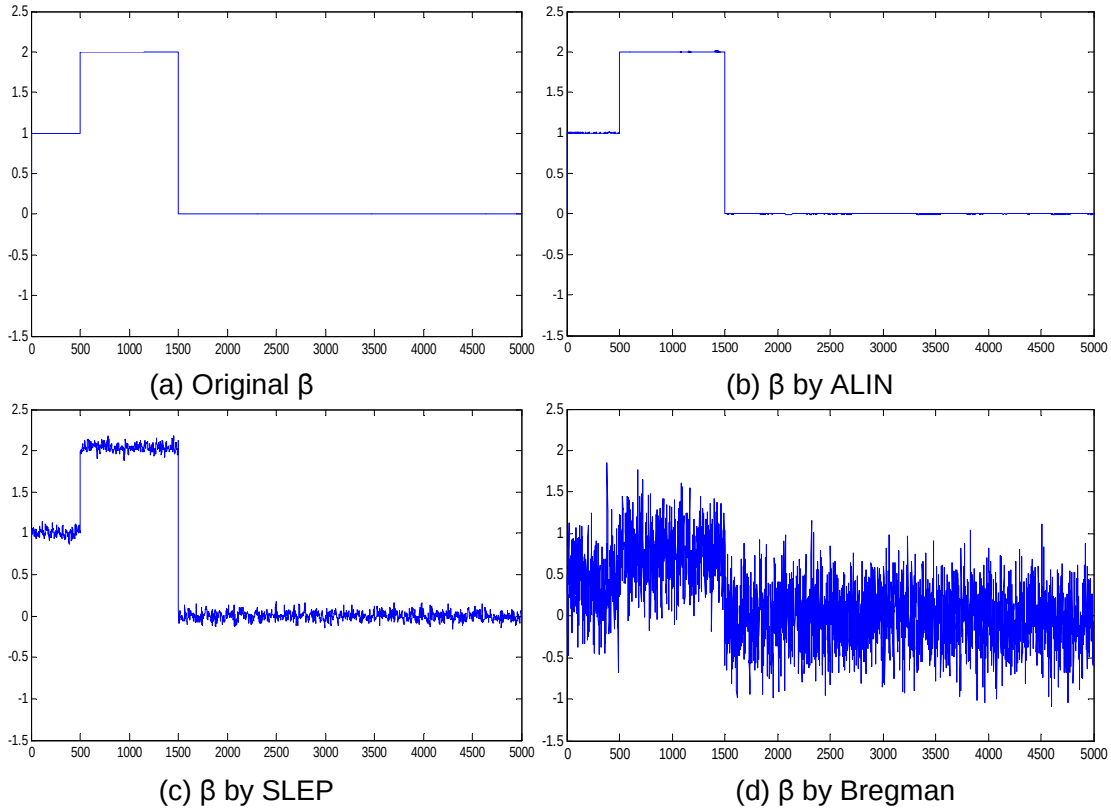


Figure 1: Results of using fused lasso penalty on a simulated data set with $n = 1000$, $p = 5000$, $\lambda = 0.1$. Plots (a), (b), (c), and (d) correspond to the original β, results from ALIN, SLEP, and Bregman, respectively.

solving the subproblems. It employs sparse data structures and converges rapidly. Usually, between 10 and 20 iterations of the conjugate gradient method are sufficient to find the solution of a subproblem.

The update test (9) is an essential element of the ALIN method. For example, in a case with $n = 1000$, $p = 5000$, and $\lambda = 0.1$, the update of $\hat{\beta}$ occurred in about 80% of the total of 70 iterations, while other iterations consisted only of improving alternating linearizations. If we allow updates of $\hat{\beta}$ at every step, the algorithm takes more than 5000 iterations to converge in this case. Similar behavior was observed in all other cases. These observations clearly demonstrate the difference between the alternating linearization method and the operator splitting methods.

5.2 CGH data example

In this study we present the results on analyzing the CGH data using fused lasso penalty. CGH is a technique for measuring DNA copy numbers of selected genes on the genome. The CGH array experiments return the

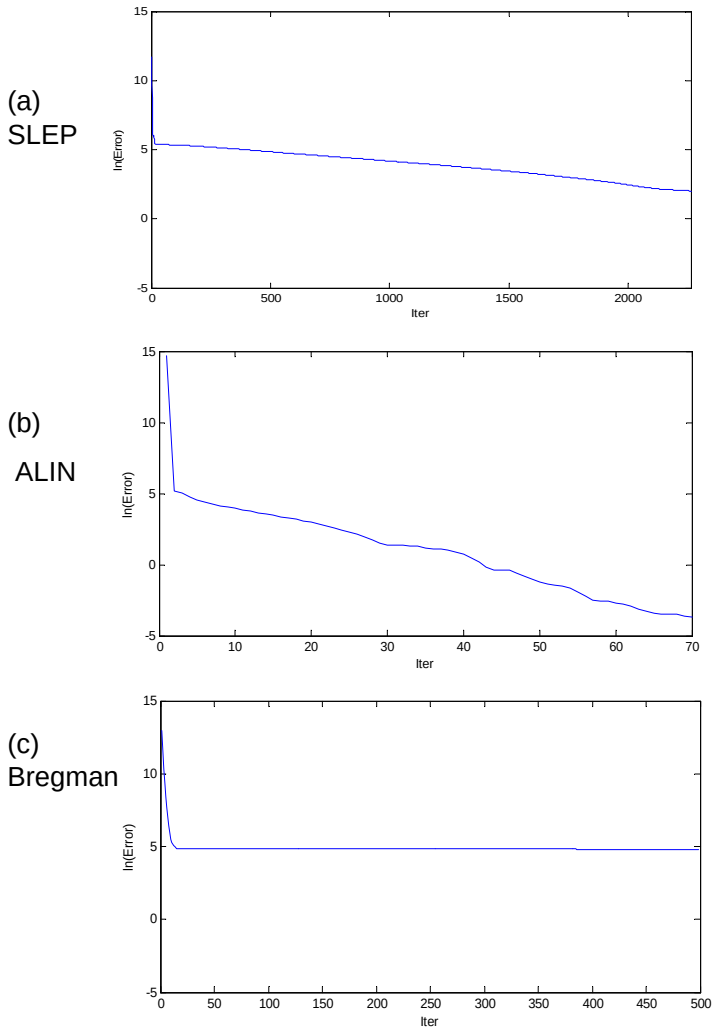
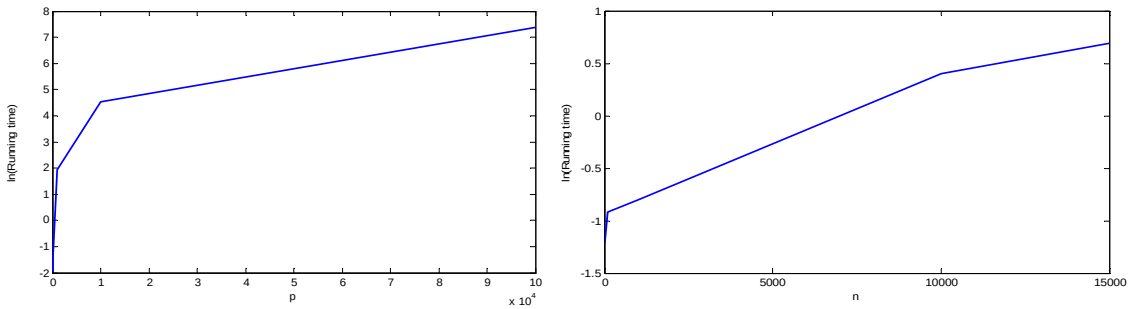


Figure 2: Simulated data set with $n = 1000$, $p = 5000$, λ=0.1. Plots (a), (b), and (c): $\ln(\text{Error})$ versus iteration number of SLEP, ALIN, and Bregman, respectively.



(a) Running time as the number of attributes changes

(b) Running time as the number of observations changes

Figure 3: Running time of ALIN as dimensions change.

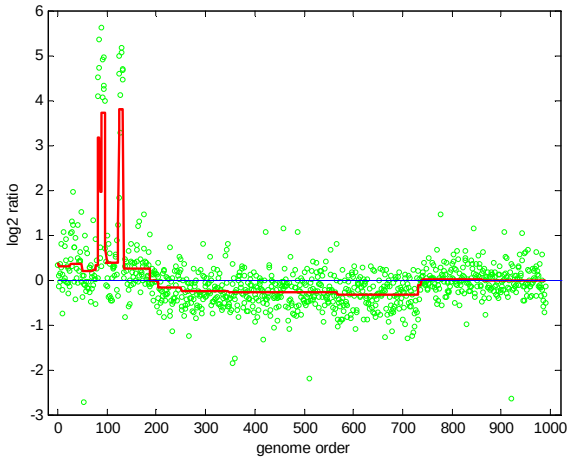


Figure 4: Fused lasso applied to CGH data, $\lambda = 3$.

log ratio between the number of DNA copies of the gene in the tumor cells and the number of DNA copies in the reference cells. A value greater than zero indicates a possible gain, while a value less than zero suggests possible losses. Tibshirani and Wang (2008) applied the fused lasso signal approximator for detecting such copy number variations.

This is a simple one-dimensional signal approximation problem with the design matrix X being the identity matrix. Thus the advantage of ALIN over the other three methods is not significant, due to the overhead that ALIN has during the conjugate gradient method implemented in MATLAB. Indeed the solution time of ALIN is comparable to that of Bregman and SLEP.

Figure 4 presents the estimation results obtained by our ALIN method. The green dots shows the original CNV number, and the red line presents the fused lasso penalized estimates.

5.3 Total variation based image reconstruction

The one-dimensional fused lasso can be naturally extended to two-dimensional cases for image processing. In this section we present several experiments on image denoising and deblurring using two-dimensional fused lasso penalty. Although of similar forms, higher-order fused lasso is fundamentally different from the one-dimensional fused lasso, as the structural matrix R defined in eq. (2) is singular. This additional complication introduces considerable challenges in the path type algorithms (Tibshirani and Taylor, 2011), and additional computational steps need to be implemented to guarantee convergence.

The ALIN algorithm does not suffer from complications due to the singularity of R, because the dual problem (19) is always well-defined and has a solution. Even if the solution is not unique, (18) is still an optimal solution of the h-subproblem, and the algorithm proceeds unaffected.

Let y be an $m \times n$ observed noisy image; one attempts to minimize the following objective function:

$$\mathcal{L}(\beta) = \tfrac{1}{2}\|y - \mathcal{A}(\beta)\|^2 + \lambda h(\beta),$$

(a) the original image (b) the noisy image (c) the denoised image

Figure 5: Results of denoising using fused lasso penalty with λ=0.05. Plots (a), (b) and (c) correspond to the original image, the noisy image, and the de-noised image, respectively.

where

$$h(\beta) = \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\left(|\beta_{i,j} - \beta_{i+1,j}| + |\beta_{i,j} - \beta_{i,j+1}|\right) + \sum_{i=1}^{m-1}|\beta_{i,n} - \beta_{i+1,n}| + \sum_{j=1}^{n-1}|\beta_{m,j} - \beta_{m,j+1}|$$

and $\mathcal{A} : \mathbb{R}^{m\times n} \to \mathbb{R}^{m\times n}$ is a linear transformation. When $\mathcal{A}$ is the identity transformation, the problem is to denoise the image y. The penalty function $h(\beta)$ is similar to the total variation (TV) penalties widely used for image denoising and deblurring.

In the following experiments, we apply the fused lasso penalty to recover noisy and blurred images to their original forms. The two images considered are of size 256 by 256, which results in solving very large fused lasso problems (the matrix R has dimensions of about 262000×66000).

In the denoising example, noise drawn from the normal distribution with zero mean and standard deviation 0.02 is added to the cameraman picture. In Figure 5 we show the denoising result. Clearly, the ALIN algorithm is able to recover the original image from the noisy image fairly well. This is due to the fact that the two-dimensional fused lasso penalizes the difference between neighboring pixel values and effectively smoothes out the image and eliminates noise. As $\mathcal{A} = I$ in this case, consistent with the observations made at the end of section 4, the ALIN method solves the denoising problem in one iteration.

In the deblurring example, we first blur the image, by replacing each pixel with the average of its neighbors and itself. This operation defines the kernel operator $\mathcal{A}$ used in the loss function $\frac{1}{2}\|y - \mathcal{A}(\beta)\|^2$. Then we add noise as before. The deblurring results are shown in Figure 6, and similar effects are observed.

Neither Split Bregman, nor SLEP is directly applicable to two-dimensional fused lasso problem. The ALIN method is able to solve this problem in 2502 seconds and 9 iterations, which clearly demonstrates its scalability. We may also inspect the de-blurred image produced by the gradient based algorithm for image deblurring (FISTA). In our experiments, we run ALIN and FISTA on the blurred image with two different values of λ and we check whether FISTA can attain the same objective function value as ALIN. It turned out

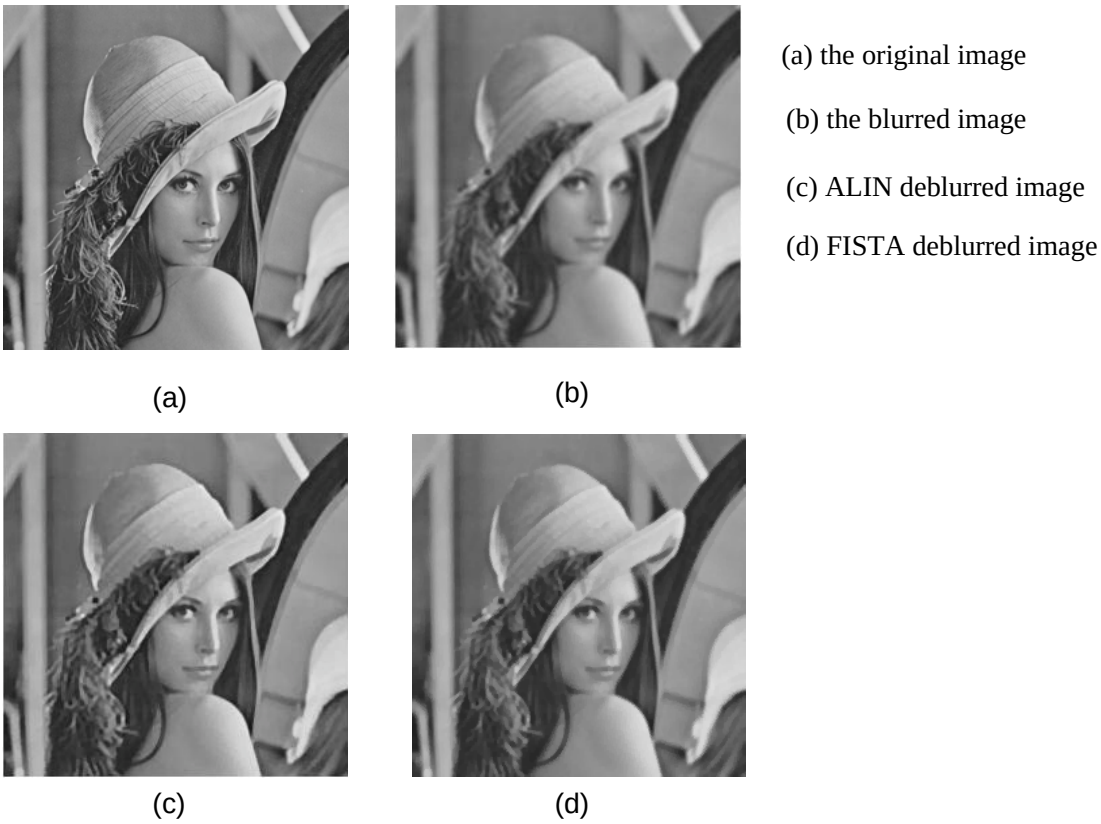


Figure 6: Results of deblurring using fused lasso penalty. Plots (a), (b), (c), and (d) correspond to the original image, the blurred image, the ALIN de-blurred image, and the FISTA d-blurred image, respectively.

that FISTA could not do so even after 10000 iterations. The results are in Table 2.

Table 2: Run time comparison on image deblurring.

Parameter	ALIN			FISTA		
	Function value	Running time	Iterations	Function value	Running time	Iterations
$\lambda = 0.001$	3.13	2800	9	5.28	2502	> 10000
$\lambda = 0.005$	11.3	2848	8	20.52	2760	> 10000

5.4 Application to a narrative comprehension study for children

With high dimensional fused lasso penalty, the constrained optimization problem with identity matrix is already difficult to solve, and a large body of literature has been devoted to solving this problem. However, there has not been a systematic treatment of penalized regression using non-separable penalty functions with an arbitrary design matrix X that may have non-full rank.

In the following we present the results of penalized regression between the measurement of children's language ability (the response y) and voxel level brain activity during a narrative comprehension task (the design matrix X). Children develop a variety of skills and strategies for narrative comprehension during early childhood years (Karunanayaka et al., 2010). This is a complex brain function that involves various cognitive processes in multiple brain regions. We are not attempting to solve the challenging neurological problem of identifying all such brain regions for this cognitive task. Instead, the goal of this study is to demonstrate ALIN's ability for solving constrained optimization problems of this type and magnitude.

The functional MRI data are collected from 313 children with ages 5 to 18 (Schmithorst et al., 2006). The experimental paradigm is a 30-second block design with alternating stimulus and control. Children are listening to a story read by adult female speaker in each stimulus period, and pure tones of 1-second duration in each resting period. The subjects are instructed to answer ten story-related multiple-choice questions upon the completion of the MRI scan (two questions per story). The fMRI data were preprocessed and transformed into the Talairach stereotaxic space by linear affine transformation. A uniform mask is applied to all the subjects so that they have measurements on the same set of voxels.

The response variable y is the oral and written language scale (OWLS). The matrix X records the activity level for all the 8000 voxels measured. The objective function is the following:

$$\mathcal{L}(\beta) = \tfrac{1}{2}\|y - X\beta\|_2^2 + \lambda h(\beta),$$

where

$$
\begin{aligned}
h(\beta) = & \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\sum_{k=1}^{p-1}\{|\beta_{i,j,k}-\beta_{i+1,j,k}|+|\beta_{i,j,k}-\beta_{i,j+1,k}|+|\beta_{i,j,k}-\beta_{i,j,k+1}|\} \\
& + \sum_{i=1}^{m-1}\sum_{k=1}^{p-1}\{|\beta_{i,n,k}-\beta_{i+1,n,k}|+|\beta_{i,n,k}-\beta_{i,n,k+1}|\} + \sum_{j=1}^{n-1}\{|\beta_{m,j,p}-\beta_{m,j+1,p}|\} \\
& + \sum_{j=1}^{n-1}\sum_{k=1}^{p-1}\{|\beta_{m,j,k}-\beta_{n,j+1,k}|+|\beta_{m,j,k}-\beta_{m,j,k+1}|\} + \sum_{i=1}^{m-1}\{|\beta_{i,n,p}-\beta_{i+1,n,p}|\} \\
& + \sum_{i=1}^{m-1}\sum_{j=1}^{n-1}\{|\beta_{i,j,p}-\beta_{i+1,j,p}|+|\beta_{i,j,p}-\beta_{i,j+1,p}|\} + \sum_{k=1}^{p-1}\{|\beta_{m,n,k}-\beta_{m,n,k+1}|\},
\end{aligned}
$$

and $m = 31$, $n = 35$, and $p = 15$.

Clearly, the design matrix does not have full rank as the dimension is far greater than the sample size. None of the methods that we compared previously work in this setting with a 3-d fused lasso penalty. The ALIN algorithm terminates in 44 steps and the run time is 3200 seconds.

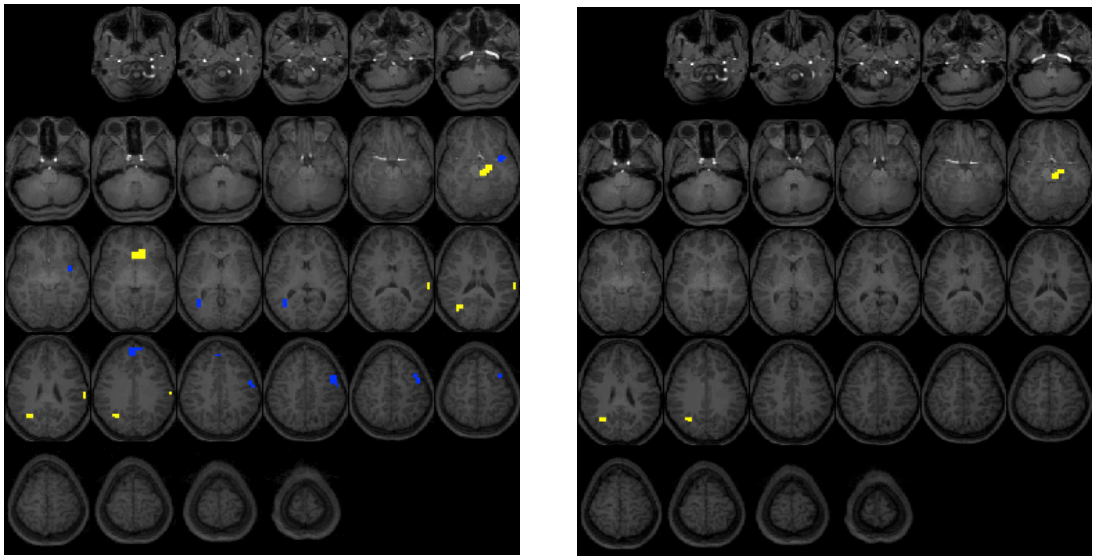


Figure 7: Results of regularization regression with combined lasso and 3-d lasso penalty. The tuning parameters of fused lasso is 0.2 for both figures. The tuning parameter for lasso is 0.2 for the left and 0.6 for the right.

While the main purpose of this study is to demonstrate the capability of the ALIN algorithm for solving penalized regression problems with 3-d lasso, there are also some interesting neurological observations. One objective of this study is to identify the voxels that are significant for explaining the performance score y. These voxels constitute active brain regions that are closely related to the OWLS. Figure 7 presents the results

of fitted coefficients using combined lasso and fused lasso penalty. The highlighted regions shown in the maps are areas with more than 10 voxels (representing clusters of size 10 and above). The left plot in the figure is the optimal solution obtained using ten-fold cross validation. The optimal tuning parameters are 0.2 for both fused lasso and lasso penalties. Roughly speaking, five brain regions have been identified. The yellow area to the rightmost side of the brain is situated in the wernicke area, which is one of the two parts of the cerebral cortex linked to speech. It is involved in the understanding of written and spoken language. The only difference between the left and right plots is the value of the tuning parameter for the lasso penalty, which is 0.2 and 0.6 respectively. Clearly, the right plot shrinks more coefficients to zero, which results in a reduced number of significant regions, as compared to the left plot.

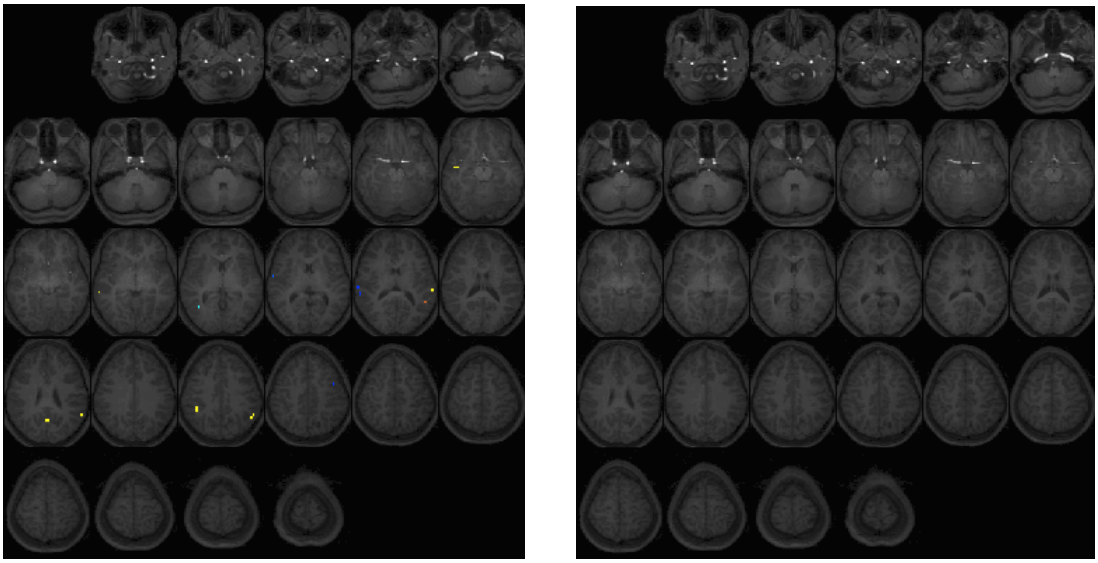


Figure 8: Results of regularization regression with lasso penalty only. The tuning parameter is 0.2. The cluster sizes are 1 and 10 for the left and right plot respectively.

We further study this regularization problem using only lasso penalty. Figure 8 shows the fitted maps. The lasso tuning parameter is 0.2 for both plots. The left plot is of cluster size 1, and the right plot uses cluster size 10. From the right plot, clearly none of the areas identified have more than 10 voxels. Comparing this with Figure 7, we see that the 3-d fused lasso penalty imposes smoothing constraints on the neighboring coefficients, thus allowing to identify larger areas significant for the response variable y. The simple lasso penalty imposes shrinkage on the coefficients individually, resulting in rather disjoint significant voxels. Such scatterness is much less informative for neurologists than larger areas identified by the three-dimensional fused lasso penalty.

6 Conclusion

The alternating linearization method is a specialized nonsmooth optimization method for solving structured nonsmooth optimization problems. It combines the ideas of bundle methods and operator splitting methods, to define a descent algorithm in terms of the values of the function that is minimized. We have adapted the alternating linearization method to structured regularization problems by introducing the idea of diagonal quadratic approximation and developing specialized methods for solving subproblems. As a result, a new general method for a variety of regularization problems has been obtained, which has the following theoretical features:

- It deals with nonsmoothness directly, not via approximations,
- It is monotonic with respect to the values of the function that is minimized,
- Its convergence is guaranteed theoretically.

Our numerical experiments on a variety of structured regularization problems illustrate the applicability of the alternating linearization method and indicate its practically important virtues: speed, scalability, and accuracy. It clearly outperforms extant methods, and it can solve problems which were unsolvable otherwise.

Its efficacy and accuracy follow from the use of the diagonal quadratic approximation and from a special test, which chooses in an implicit way the best operator splitting step to be performed. The current approximate solution is updated only if it leads to a significant decresase of the value of the objective function.

Its scalability is due to the use of highly specialized algorithms for solving its two subproblems. The algorithms do not require any explicit matrix formation or inversion, but only matrix–vector multiplications, and can be efficiently implemented with sparse data structures.

Our study of narrative comprehension for children in section 5.4 is an illustration of broad applicability of the alternating linearization method. It is, to the best of our knowledge, the first successful method for this three-dimensional fused lasso problem.

References

H. H. Bauschke and P. L. Combettes. *Convex analysis and monotone operator theory in Hilbert spaces*. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. Springer, New York, 2011.

A. Beck and M. Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2(1):183–202, 2009.

E. G. Birgin and J. M. Martínez. Large-scale active-set box-constrained optimization method with spectral projected gradients. *Comput. Optim. Appl.*, 23(1):101–125, 2002.

S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein. Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends in Machine Learning*, 3(1):1–122, 2010.

X. Chen, S. Kim, Q. Lin, J. Carbonell, and E. Xing. Graph-structured multi-task regression and an efficient optimization method for general fused lasso. technical report 1005.3579v1, arXiv, 2010.

X. Chen, S. Kim, Q. Lin, J. Carbonell, and E. Xing. An efficient proximal gradient method for general structured sparse learning. technical report 1005.4717v3, arXiv, 2011.

P. L. Combettes. Iterative construction of the resolvent of a sum of maximal monotone operators. *J. Convex Anal.*, 16(3-4):727–748, 2009.

P. L. Combettes and J.-C. Pesquet. Proximal splitting methods in signal processing. technical report 0912.3522v4, arXiv, 2010.

I. Daubechies, M. Defrise, and C. De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Comm. Pure Appl. Math*, 57(11):1413–1457, 2004.

J. Douglas and H. H. Rachford. On the numerical solution of heat conduction problems in two and three space variables. *Trans. Amer. Math. Soc.*, 82:421–439, 1956.

J. Eckstein and D. P. Bertsekas. On the Douglas-Rachford splitting method and the proximal point algorithm for maximal monotone operators. *Math. Programming*, 55(3, Ser. A):293–318, 1992.

B. Efron, T. Hastie, I. Johnstone, and R. Tibshirani. Least angle regression. *Annals of Statistics*, 32(2): 407–451, 2004.

J.M. Fadili and G. Peyré. Total variation projection with first order schemes. *IEEE Transactions on Image Processing*, 20(3):657 –669, 2011.

A. Friedlander and J. M. Martínez. On the maximization of a concave quadratic function with box constraints. *SIAM J. Optim.*, 4(1):177–192, 1994.

J. Friedman, T. Hastie, H. Hoefling, and R. Tibshirani. Pathwise coordinate optimization. *Annals of Applied Statistics*, 1(2):302–332, 2007.

W. Fu. Penalized regressions: the bridge vs. the lasso. *Journal of Computational and Graphical Statistics*, 7(3):397–416, 1998.

T. Goldstein and S. Osher. The split Bregman method for $L1$-regularized problems. *SIAM J. Imaging Sci.*, 2(2):323–343, 2009.

G. H. Golub and C. F. Van Loan. *Matrix Computations*. Johns Hopkins Studies in the Mathematical Sciences. Johns Hopkins University Press, Baltimore, MD, third edition, 1996.

B. He and X. Yuan. On the $O(1/t)$ convergence rate of alternating direction method. technical report, Optimization On-Line, 2011.

J.-B. Hiriart-Urruty and C. Lemaréchal. *Convex analysis and minimization algorithms. II*, volume 306 of *Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]*. Springer-Verlag, Berlin, 1993.

H. Hoefling. A path algorithm for the fused lasso signal approximator. *Journal of Computational and Graphical Statistics*, 19(4), 2010.

P. Karunanayaka, V. J. Schmithorst, J. Vannest, J. P. Szaflarski, E. Plante, and S. K. Holland. A group independent component analysis of covert verb generation in children: A functional magnetic resonance imaging study. *Neuroimage*, 51:472–487, 2010.

K. Kiwiel, C. Rosa, and A. Ruszczyński. Proximal decomposition via alternating linearization. *SIAM Journal on Optimization*, 9:153–172, 1999.

K. C. Kiwiel. *Methods of descent for nondifferentiable optimization*, volume 1133 of *Lecture Notes in Mathematics*. Springer-Verlag, Berlin, 1985.

P.-L. Lions and B. Mercier. Splitting algorithms for the sum of two nonlinear operators. *SIAM J. Numer. Anal.*, 16(6):964–979, 1979.

J. Liu, L. Yuan, and J. Ye. An efficient algorithm for a class of fused lasso problems. *ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, 2010.

J. Liu, L. Yuan, and J. Ye. SLEP: Sparse learning with efficient projections. techincal report, Computer Science Center, Arizona State University, 2011.

Yu. Nesterov. Gradient methods for minimizing composite objective function. discussion paper 2007/76, CORE, 2007.

D. W. Peaceman and H. H. Rachford. The numerical solution of parabolic and elliptic differential equations. *J. Soc. Indust. Appl. Math.*, 3:28–41, 1955.

A. Ruszczyński. On convergence of an augmented Lagrangian decomposition method for sparse convex optimization. *Math. Oper. Res.*, 20(3):634–656, 1995.

A. Ruszczyński. *Nonlinear optimization*. Princeton University Press, Princeton, NJ, 2006.

V. Schmithorst, S. Holland, and E. Plante. Cognitive modules utilized for narrative comprehension in children: a functional magnetic resonance imaging study. *Neuroimage*, 29:254–266, 2006.

R. Tibshirani. Regression shrinkage and selection via the lasso. *Journal of The Royal Statistical Society Series B*, 58(1):267–288, 1996.

R. Tibshirani and J. Taylor. The solution path of the generalized lasso. *Annals of Statistics*, 2011.

R. Tibshirani and P. Wang. Spatial smoothing and hot spot detection for cgh data using the fused lasso. *Biostatistics*, 9(1):18–29, 2008.

R. Tibshirani, M. Saunders, J. Zhu, and S. Rosset. Sparsity and smoothness via the fused lasso. *Journal of The Royal Statistical Society Series B*, 67:91–108, 2005.

P. Tseng. Convergence of block coordinate descent method for nondifferentiable minimization. *Journal of Optimization Theory and Applications*, 109:474–494, 2001.

G.-B. Ye and X. Xie. Split Bregman method for large scale fused Lasso. *Comput. Statist. Data Anal.*, 55(4): 1552–1569, 2011.

Appendix

Preconditioned conjugate gradient method

The preconditioned conjugate gradient method is used in our method to solve quadratic optimization problems of the following form:

$$\min_{x\in\bar{F}} \; f(x) = \tfrac{1}{2}x^T Ax - b^T x,$$

where $x \in \mathbb{R}^n$, A is a positive semidefinite matrix of dimension $n \times n$, $b \in \mathbb{R}^n$, and $\bar{F}$ is a box in $\mathbb{R}^n$.

In the f-subproblem (14), x corresponds to the difference $\delta = \beta - \hat{\beta}$, $A = X^T X + \mathrm{diag}(X^T X)$ and is positive definite, $b = X^T(y - X\hat{\beta}) - s_h$, and $\bar{F}$ is the whole space. In this case we use the preconditioner $M = 2\,\mathrm{diag}(X^T X)$. The method always starts from $x^0 = 0$, which corresponds to $\beta = \hat{\beta}$.

In the dual problem (19) of the h-subproblem, x corresponds to the multipliers μ, $A = RD^{-1}R^T$, $b = R(\hat{\beta} - D^{-1}s_f)$ and $\bar{F}$ is a closed face of the set

$$\Omega = \{x \in \mathbb{R}^n : \|x\|_\infty \le \lambda\}. \tag{22}$$

We use the preconditioner $M = \mathrm{diag}(RD^{-1}R^T)$. The method is a part of a more complicated algorithm to be described in the next section. It operates on a subspace of the space of multipliers, that is, x is a subvector

of μ, and the other matrices and vectors are corresponding submatrices and subvectors of A, b, and M. We still use the same notation; it will not lead to misunderstandings, we hope. The method starts from 0 or from the best point found so far.

Algorithm 2 Preconditioned Conjugate Gradient Method

1: $g^0 \leftarrow Ax^0 - b$, $z^0 \leftarrow M^{-1}g^0$, $d^0 \leftarrow -z^0$, $k \leftarrow 0$
2: **repeat**
3: $\quad \tau_k \leftarrow \dfrac{(g^k)^T z^k}{(d^k)^T A d^k}$
4: $\quad x^{k+1} \leftarrow x^k + \tau_k d^k$
5: $\quad$ **if** $(x^{k+1} \notin \bar{F})$ **then**
6: $\quad\quad \tau_k \leftarrow \max\{\tau : x^k + \tau d^k \in \bar{F}\}$
7: $\quad\quad x^k \leftarrow x^k + \tau_k d^k$
8: $\quad\quad$ **return**
9: $\quad$ **end if**
10: $\quad g^{k+1} \leftarrow g^k + \tau_k A d^k$
11: $\quad z^{k+1} \leftarrow M^{-1} g^{k+1}$
12: $\quad \alpha_k \leftarrow \dfrac{(z^{k+1})^T (g^{k+1} - g^k)}{(z^k)^T g^k}$
13: $\quad d^{k+1} = -z^{k+1} + \alpha_k d^k$
14: $\quad k \leftarrow k + 1$
15: **until** { $(\|g^k\| \leq \varepsilon)$ **or** (*Leave Face*) }

If the method is applied to an unconstrained problem and satisfies the stopping test in line 15, then the point x^k is an approximation of the solution of the problem. If the stopping test $\|g^k\| \leq \varepsilon$ is satisfied in the box-constrained problem, the optimal solution in the current face $\bar{F}$ is found. The condition *Leave Face* verifies whether it is beneficial to leave the current face $\bar{F}$ before reaching optimality. We describe it in more detail in the next section. In general, further operations are performed to change the face over which optimization is performed, or to detect optimality. If the method reaches the boundary of the face $\bar{F}$ in line 5 and returns, other operations are performed to change the face over which optimization is performed. In both cases, the point x^k becomes the starting point for the next phase of the method, and the index k is reset to 0. We also reset the method in the case when $(g^k)^T z^k \leq 0$.

Active-set box-constrained algorithm with spectral projected gradients

Active-set box-constrained algorithm (GENCAN) minimizes the function $f(x)$ on the box Ω defined in (22). The feasible region Ω is divided into disjoint faces by specifying active constraints for each face. For each partition $\mathit{\Pi} = \{I_-, I_0, I_+\}$ of the set $I = \{1, 2, \ldots, n\}$ into three disjoint sets, so that $I = I_- \cup I_0 \cup I_+$, we specify the corresponding face as

$$F^{\mathit{\Pi}} = \{x \in \Omega : x_i = -\lambda \text{ if } i \in I_-,\ x_i = \lambda \text{ if } i \in I_+,\ -\lambda < x_i < \lambda \text{ if } i \in I_0\}.$$

Within the current face we run the conjugate gradient method, as specified above, by replacing at each point $x \in F^{\Pi}$ the gradient $g = \nabla f(x)$ by the "gradients within the face":

$$g_i^{\Pi} = \begin{cases} 0 & \text{if } i \in I_- \cup I_+, \\ g_i & \text{if } i \in I_0, \end{cases} \qquad i = 1, \ldots, n.$$

If the conjugate gradient method reaches the boundary of the current face F and exits in line 8, then one or more indices $i \in I_0$ are moved to either I_- or I_+, depending on whether $x_i^k = -\lambda$ or $x_i^k = \lambda$. After that, the current point x^k becomes the starting point x^0 for the next pass of the conjugate gradient method in the new face.

At the end of each iteration of the conjugate gradient method we verify the following *Leave Face* test. The "projected gradient" at $x \in \Omega$ is defined as:

$$g_i^P = \begin{cases} 0 & \text{if } i \in I_- \text{ and } g_i > 0 \text{ or } i \in I_+ \text{ and } g_i < 0, \\ g_i & \text{otherwise,} \end{cases} \qquad i = 1, \ldots, n.$$

If $\|g^P\| = 0$, then the optimal solution has been found. Otherwise, if

$$\|g^{\Pi}\| \leq \eta \|g^P\|, \tag{23}$$

where $\eta \in (0, 1)$ is a fixed constant, then we decide that it is beneficial to leave the face F (the *Leave Face* test is true). In the latter case, for a "spectral coefficient" $\sigma_k > 0$, we calculate the direction

$$d^k = P_{\Omega}(x^k - \sigma_k g) - x^k,$$

where $P_{\Omega}(\cdot)$ is the operation of the orthogonal projection on Ω. The calculation of σ_k, which measures the curvature of the objective function along the last direction, is essential here; in a typical case

$$\sigma_k = \frac{(x^k - x^{k-1})^T (g^k - g^{k-1})}{\|x^k - x^{k-1}\|^2}.$$

Next, having the current point x^k and the direction d^k, we carry out the constrained line search:

$$\tau_k = \arg\min\{f(x^k + \tau d^k) : \tau \geq 0,\ x^k + \tau d^k \in \Omega\}, \qquad x^{k+1} = x^k + \tau_k d^k.$$

Due to condition (23), this step will always result in the change of the current face. After that, the new face F is determined, and the conjugate gradient method re-starts from the point x^{k+1}, which plays the role of the point x^0 in the new cycle.

More details about GENCAN can be found in (Birgin and Martínez, 2002; Friedlander and Martínez, 1994).